CLAIMS

What is claimed is:

1. A method of routing data packets between nodes in a wireless network, comprising:

receiving data packet traffic for a destination node;

dynamically tracing a route to said destination node in response to receipt of said traffic if no suitable route is found in the routing table;

loop-checking the complete path prior to entering said dynamically traced route into said routing table; and

transmitting said traffic to said destination node according to said routing table.

2. A method as recited in claim 1, wherein a data packet received for transmission comprises a header with source and destination information.

3. A method as recited in claim 2, wherein said header does not contain a sequence number, or equivalent, associated with the destination node.

4. A method as recited in claim 1:

wherein said dynamic tracing obtains information about the length and second-to-last hop of the shortest path to all known destinations,

whereby counting to infinity problems are avoided.

5. A method as recited in claim 1, wherein entries in said routing table comprise:

an entry for each known destination;

wherein each entry has a destination identifier j;

a successor to said destination, s_j^i;

a second-to-last hop to said destination p_j^i;

distance to said destination D_j^i; and

a route tag tag_j^i.

6. A method as recited in claim 5, wherein the route tag may contain a value selected from the group of route values consisting essentially of *correct*, *null*, *error*, or equivalents, which indicate the status of the route to which said entry is associated.

7. A method as recited in claim 5, wherein a distance table is associated with said routing table and comprises:

a matrix is distance values D_{jk}^i of the route from i to j through k; and

an entry for the second-to-last hop p_{jk}^i on that route.

8. A method as recited in claim 1, wherein routing links to a given neighbor are discovered only in response to traffic being received for destination for which no

correct route exists in the routing table.

9. A method as recited in claim 1, wherein said dynamic tracing of routes to the destination is performed by sending *Query* commands to discover routing information from neighboring nodes.

10. A method as recited in claim 9, wherein a query table is maintained to controls the extent to which a query is forwarded.

11. A method as recited in claim 10, wherein the extent of forwarding is controlled by tracking the number of hops the query has made from the sender in relation to a forwarding limit.

12. A method as recited in claim 11, wherein the forwarding limit comprises a predetermined maximum hop count values, MAX_HOPS, or equivalent.

13. A method as recited in claim 1, wherein links to neighboring nodes are only discovered in response to the receipt of an *Update* or *Query* control packet from that neighbor.

14. A method as recited in claim 1, wherein said protocol provides for packet routing without the use of a link-layer protocol for monitoring link connectivity with

neighbors.

15. A method for routing data packets in a wireless network at a node *i*, comprising:

maintaining a routing table of one or more known neighbors along with link cost to said known neighbors;

performing loop checking of complete paths prior to an entry being made into the routing table; and

broadcasting a routing message from said node;

said routing message comprising a vector of entries;

wherein each entry in said vector of entries corresponds to a route in the routing table; and

wherein each said entry in said vector of entries contains a destination identifier j, the distance to the destination D_j^i, and the second-to-last hop to that destination p_j^i.

16. A method as recited in claim 15:

wherein a first node considers a second as its neighbor if it hears update messages from said second node; and

wherein said first node no longer considers said second node as its neighbor if said first node cannot send data packets to said second node.

17. A method as recited in claim 15, wherein said routing table contains entries for all known destinations with each entry comprising a destination identifier j, the successor to that destination s_j^i, the second-to-last hop to the destination p_j^i, the distance to the destination D_j^i and a route tag tag_j^i.

18. A method as recited in claim 17:

wherein when the element tag_j^i is set to a value of $Correct$, it implies a loop-free finite value route;

wherein when element tag_j^i set to $Null$ it implies that the route still remains to be checked; and

wherein when the element tag_j^i is set to $Error$ an infinite metric route, or a route with a loop, is implied.

19. A method as recited in claim 18, further comprising:

maintaining a distance table at said node;

wherein said distance table at router i comprises a matrix of distance values of the route from i to j through k, D_{jk}^i and the second-to-last hop p_{jk}^i on that route.

20. A method as recited in claim 19, wherein D_{jk}^i is set to $RD_j^k + l_k^i$ where RD_j^k is the distance reported by k to j in the last routing message and l_k^i is the cost of link

(i,k).

21. A method as recited in claim 20, wherein said link cost is a function of hop count.

22. A method as recited in claim 20, wherein said link cost is a function of latency.

23. A method as recited in claim 20, wherein said link cost is a function of bandwidth.

24. A method for routing data packets in a wireless network at a node *i*, comprising:

maintaining a routing table of one or more known neighbors along with link cost to said known neighbors;

routing data packets based on entries in said routing table;

wherein said routing table contains entries for all known destinations;

each said entry in said routing table comprising

a destination identifier j,

the successor to said destination s_j^i,

the second-to-last hop to the destination p_j^i,

distance to the destination D_j^i, and

a route tag tag_j^i.

25. A method as recited in claim 24,

wherein when the element tag_j^i is set to a value of $Correct$, it implies a loop-free finite value route,

wherein when the element tag_j^i set to $Null$ it implies that the route still remains to be checked, and

wherein when the element tag_j^i is set to $Error$ an infinite metric route, or a route with a loop, is implied.

26. A method as recited in claim 25, further comprising:

maintaining a distance table at said node;

wherein said distance table at router i comprises a matrix of distance values of the route from i to j through k, D_{jk}^i and the second-to-last hop p_{jk}^i on that route.

27. A method as recited in claim 26, wherein D_{jk}^i is set to $RD_j^k + l_k^i$ where RD_j^k is the distance reported by k to j in the last routing message and l_k^i is the cost of link (i,k).

28. A method as recited in claim 27, wherein said link cost is a function of hop count.

29. A method as recited in claim 27, wherein said link cost is a function of latency.

30. A method as recited in claim 27, wherein said link cost is a function of bandwidth.

31. A method for routing data packets in a wireless network at a node i, comprising:

creating a route for a destination j only when a data packet for j arrives by,

(i) broadcasting a query out to all neighbors;

(ii) forwarding node will forward a query to all its neighbors only if it does not have a route to the destination j and if the following conditions are met:

(a) the number of hops query packet has already been forwarded by < *MAX_HOPS,*

(b) it has been greater than *query_receive_timeout* since the last query forwarded for that destination,

whereby only local clocks used for query_recv_timeouts,

(iii) broadcasting back an update instead of forwarding a query if a route to destination j exists and the route value to i decreases from infinite to finite after

processing the query,

(iv) utilizing rules in step (iii) to forward an update back to the i node,

(iv) wherein when the update reaches i, i has a route to j.

32. A method for Maintaining a route to a destination, comprising

selecting a neighbor *p* as the next hop in a route from node *i* to destination j if,

(i) the path from neighbor *p* to destination *j* does not include node i and does not repeat any node, and $D^i_{yp} < D^i_{yx}$,

(ii) for any other neighbor *x* and for all nodes *y* that are in the path from destination j to neighbor *p*, $D^i_{yp} > D^i_{yx}$,

wherein the distance value of the route from node i to node *y* through neighbor *p* is the distance value of the route from node i to node *y* through neighbor x.

33. A method as recited in claim 32, further comprising:

sending updates to a routing table if either of the following conditions are met,

(i) a node loses the last path to a destination, or

(ii) a node suffers a distance increase to a destination.